

# Weekly Report(2018.10.22-2018.10.29)

---

## DONE

---

### 事项

- D3.js学习

在Udacity上学习了课程\*\*《数据可视化和D3.js基础》，\*\* lesson6~lesson7，学习了数据可视化设计原则中的叙事结构，学习了 D3.js 的数据绑定，完成了一个 D3.js 的大型样例，源码及其效果附在报告文末。

- 张玮老师国庆时布置的任务

看过了[精选 | 超全可视化课程合集，助你C位出道](#)中剩下的可视化课程，基本都要收费，因此打算周一就上传 Udacity 中的两节网课的视频和课程资料到小组FTP中。

- 看论文TCPTree

继续看论文，看了第二部分的时间相关变量TCV空间和分析任务，以及使用到的信息论的公式。

- 其他

学习了 viki 语法，布置了 vagwiki 的周报主页

### 小结

本周搞清楚了 D3.js 中数据绑定的知识点，记录在笔记上；论文TCPTree继续在看。  
本周学习时间不长，主要在看网课和实现例子上。

## PLAN

---

### 短期计划

- 周一、周二待在实验室，周三、周四本校有事，周五之后尽量多来
- 继续看论文TCPTree
- 开始看论文TPFlow
- 开始学习 Node.js

### 中期计划

- 能在11月初和学姐交流想法
- 学习服务端知识，更新个人网站

## 长期计划

- 跟紧VIS2019的项目进度
- 能用可视化的先进工具实现自己的idea

# Appendix

## D3.js Demo

- win10环境下cmd打开运行窗口，在目标文件夹输入 `python -m http.server`

› 此电脑 › Windows (C:) › 用户 › 14403 › 桌面 › D3

名称	修改日期	类型	大小
 L3_basic_charts_final.html	2018/10/26 22:31	Chrome HTML Doc...	6 KB
 world_cup_geo.tsv	2014/10/12 11:14	TSV 文件	190 KB

 命令提示符 - `python -m http.server`

```
Microsoft Windows [版本 10.0.17134.345]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\14403>cd desktop

C:\Users\14403\Desktop>cd D3

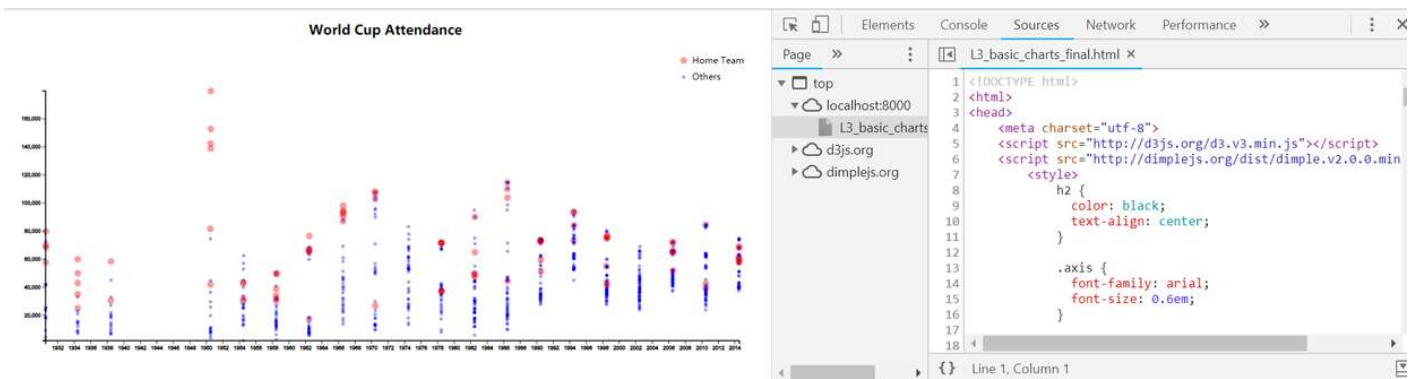
C:\Users\14403\Desktop\D3>python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

- 在Chrome中访问 `http://localhost:8000`

# Directory listing for /

- [L3\\_basic\\_charts\\_final.html](#)
- [world\\_cup\\_geo.tsv](#)

- 运行该html文件效果如下



数据是所有世界杯中每场比赛的日期和观赛人数，其中红色圆圈代表当年的世界杯东道主的比赛，蓝色圆圈则是其他队伍的比赛。

观察图表，在世界杯早期，交通不通畅的时候，明显的是东道主的比赛观赛人数最多，当交通发达后，部分热门球队的比赛就超过了东道主。

- 源码

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <script src="http://d3js.org/d3.v3.min.js"></script>
  <script src="http://dimplejs.org/dist/dimple.v2.0.0.min.js"></script>
  <style>
    h2 {
      color: black;
      text-align: center;
    }

    .axis {
      font-family: arial;
      font-size: 0.6em;
    }

    path {
      fill: none;
      stroke: black;
      stroke-width: 2px;
    }

    .tick {
      fill: none;
      stroke: black;
    }

    circle {
      opacity: 0.4;
      stroke: none;
    }

    .line_plot {
      fill: none;
      stroke: #4eb0bb;
      stroke-width: 1px;
    }
  </style>
  <script type="text/javascript">
    //用于日期的处理, in:27-05-1934 (16:00 h) out:一个JS对象
    format = d3.time.format("%d-%m-%Y (%H:%M h)");
    //D3处理数据绘制图表的主函数
    function draw(data) {
      //设定参数
      "use strict";
      var margin = 75, //图表的外间距
          width = 1400 - margin, //图表的宽度
          height = 600 - margin, //图表的高度
          var radius = 3, //圆圈的半径
          multiplier = 2; //大圆圈的放大倍率

      //选择body, 添加标题
      d3.select("body")
        .append("h2")
        .text("World Cup Attendance");
  </script>

```

```

//选择body, 添加svg, 设置宽度高度, 添加group, 设置为图表
var svg = d3.select("body")
.append("svg")
.attr("width", width + margin)
.attr("height", height + margin)
.append('g')
.attr('class', 'chart');

/*
Dimple.js 图表函数, 这里使用D3来实现
*/

//选择svg, 选择所有circle元素, 数据绑定后enter未显示的数据, 添加circle
d3.select('svg')
.selectAll("circle")
.data(data)
.enter()
.append("circle")

//区间函数, 获取date的最小值和最大值
var time_extent = d3.extent(data, function(d) {
return d['date'];
});
//尺度映射函数, 将margin-width和data的最值——对应 (类型是time)
var time_scale = d3.time.scale()
.range([margin, width])
.domain(time_extent);

//区间函数, 获取attendance的最小值和最大值
var count_extent = d3.extent(data, function(d) {
return d['attendance'];
});
//尺度映射函数, 将height-margin和count的最值——对应 (类型是linear)
var count_scale = d3.scale.linear()
.range([height, margin])
.domain(count_extent);

//创建axis轴 对象, 设定ticks区间为2years
var time_axis = d3.svg.axis()
.scale(time_scale)
.ticks(d3.time.years, 2);
//选择svg, 新建group, 设定属性是x-axis, 移动: 起始点为0,height,call表达为time轴
d3.select("svg")
.append('g')
.attr('class', 'x axis')
.attr('transform', "translate(0," + height + ")")
.call(time_axis);

//创建axis轴 对象, 设定orient数字方向在left
var count_axis = d3.svg.axis()
.scale(count_scale)
.orient("left");
//选择svg, 新建group, 设定属性是y-axis, 移动: 起始点为margin,0,call表达为count轴
d3.select("svg")
.append('g')

```

```

.attr('class', 'y axis')
.attr('transform', "translate(" + margin + ",0)")
.call(count_axis);

//选择所有circle, 设定cx, 设定cy, 设定r (东道主更大), 设定filled (东道主红色)
d3.selectAll('circle')
.attr('cx', function(d) {
    return time_scale(d['date']);
})
.attr('cy', function(d) {
    return count_scale(d['attendance']);
})
.attr('r', function(d) {
    if (d['home'] === d['team1'] || d['home'] === d['team2']) {
        return radius * multiplier;
    } else {
        return radius;
    }
})
.attr('fill', function(d) {
    if (d['home'] === d['team1'] || d['home'] === d['team2']) {
        return 'red'
    } else {
        return 'blue';
    }
});

//添加group, 类型为Legend图例, 设定起始点, 选择所有group元素, 并数据绑定新的data
//使用enter选择所有未显示的元素->这一步完成以后其实就只变成了这个未出现的g
var legend = svg.append("g")
.attr("class", "legend")
.attr("transform", "translate(" + (width - 100) + "," + 20 + ")")
.selectAll("g")
.data(["Home Team", "Others"])
.enter().append("g");
//设定Legend (是一个group) 的子元素circle
legend.append("circle")
.attr("cy", function(d, i) {
    return i * 30;
})
.attr("r", function(d) {
    if (d === "Home Team") {
        return radius * multiplier;
    } else {
        return radius;
    }
})
.attr("fill", function(d) {
    if (d === "Home Team") {
        return 'red';
    } else {
        return 'blue';
    }
});
//设定Legend (是一个group) 的子元素text
legend.append("text")
.attr("y", function(d, i) {

```

```

        return i * 30 + 5;
    })
    .attr("x", radius * 5)
    .text(function(d) {
        return d;
    });
};
</script>
</head>

<body>
  <script type="text/javascript">
    /*
    Use D3 (not dimple.js) to load the TSV file
    and pass the contents of it to the draw function
    */
    //d3获取数据文件, 将其传至draw文件, 其中可以使用匿名的预处理函数
    //该函数使得: attendance从string变成number, 使用+;
    //data从string变成给定的js time object
    d3.tsv("world_cup_geo.tsv", function(d) {
        d['attendance'] = +d["attendance"];
        d['date'] = format.parse(d['date']);
        return d;
    }, draw);
  </script>
</body>
</html>

```